SÉBASTIEN DUDEK

# Serveurs Dédiés Protégez-vous des attaques sur le Web

Degré de difficulté



Les serveurs sont les principales cibles d'attaques sur le Web et pourtant les questions au sujet de la sécurité sont souvent laissées de côté, ce qui amène très souvent à des conséquences catastrophiques.

Les entreprises comme les particuliers ont des besoins très précis concernant le stockage et la publication des informations. Il est clair qu'utiliser un serveur mutualisé serait trop risqué pour héberger du contenu confidentiel et des projets en cours de développement.

En effet, sur les mutualisés certains droits d'accès sont oubliés ou mal configurés. Chaque projet se retrouvant sur le même serveur et il faut limiter l'exécution des scripts à leur répertoire respectifs avec le *Safe_Mod* ou *Open_Base_Dir* (largement conseillé). Seulement, l'utilisation de ces limitations est parfois rendue obsolète par *bypassing* qui consiste à exploiter une faille fonctionnelle présente dans un langage puissant tel que PHP, Perl, Python. Même si une série de patches est appliquée pour *boucher les trous* de sécurité, il faut garder à l'esprit que les correctifs au niveau de la sécurité ne peuvent que complexifier une attaque. C'est pour cela que vous tendrez à utiliser un serveur dédié afin de marquer une séparation avec les autres utilisateurs.

Cela nous amène tout de même à nous poser la question suivante : nos projets séparés des autres utilisateurs sont-ils réellement en sécurité ?

Pour y répondre, nous verrons en parallèle les différents types d'attaques avec les correctifs à appliquer pour éviter les principales attaques.

## La visibilité

La visibilité, contrairement à une idée répandue, est une arme pour l'attaquant qui connaîtra ainsi les services utilisés avec leurs versions et leurs vulnérabilités référencées ou après analyse des codes sources. Le problème des serveurs par défaut est qu'ils sont toujours visibles en laissant tourner des services ou en répondant aux *pings*.

Être invisible, sur le web bien évidement, est chose possible si le serveur fonctionne en passif, ne répondant à aucune sonde ni enquête. Il est judicieux de bien choisir les vecteurs visibles ou non par le système. En effet, les services en externe ne devant pas nécessairement être exposés peuvent simplement rester en interne.

### Transfert de zone : The Old Fashion brute force

Vous remarquez qu'après avoir restreint le transfert de zone, aucun résultat n'est retourné mais il faut avoir à l'esprit que ce n'est qu'une question de temps car d'autres techniques subsistes. En effet, comme la recherche de mot de passe, il est possible d'effectuer une attaque par brute force.

Nous pourrions imaginer un algorithme qui prend un nom de domaine et teste une série d'alias enregistrés dans un dictionnaire. Et après avoir vérifié le statut de ces alias, nous pouvons en rechercher d'autres suivant une grammaire précise de [a-z0-9] par combinaison.

Un tel algorithme existe sous le nom de *Fierce* et est disponible à cette adresse : *http://ha.ckers.org/fierce/*

### CET ARTICLE EXPLIQUE...

La prise d'information.

L'identification des vecteurs d'attaques et des vulnérabilités

Les différents types d'attaques après analyse des informations recueillies.

La sécurisation des services.

### CE QU'IL FAUT SAVOIR...

Administration de serveurs (Unix, Linux, Windows, ...)

Des notions en penetration testing.

Bases de survie en programmation (C, PHP, Perl, Python, ...)

## Connaître sa cible

Avant la prise d'information, le serveur cible est considéré comme une sorte de *blackbox* car nous ne connaissons rien de notre victime, à part son adresse *opihad.com* (fictive pour l'exemple).

La consultation de bases de données WHOIS commence à être connue de tous, même chez les avocats, secrétaires et autres personnes n'ayant pas forcément les connaissances techniques en sécurité informatique. Lorsque nous achetons un nom de domaine, les informations sur l'acheteur ainsi que le ou les responsables techniques (nom, prénom, adresse, code postal, numéro de téléphone), les serveurs DNS principaux (dns1.opihad.com, dn2.opihad.com, dn3.opihad.com).

**Listing 1.** *Résultat de notre WHOIS pour opidah.com (parties intéressantes)*

```
wwitb~# whois opidah.com
domain: opidah.com
reg_created: 2009-04-22 13:06:25
expires: 2011-04-22 13:06:25
created: 2009-04-22 15:06:26
changed: 2010-11-12 15:38:03
transfer-prohibited: yes
ns0: dns1.opidah.com
ns1: dns2.opidah.com
ns2: dns3.opidah.com
owner-c:
  nic-hdl: TG8520-DIGAN
  owner-name: James Paledown
  organisation: ~
  person: James Paledown
 address: '12, place des zics'
  zipcode: 75014
  city: Paris
  country: France
  phone: +33.145147005
  fax: ''
  email: admin@opidah.com
admin-c:
  nic-hdl: TG8520-DIGAN
  owner-name: James Paledown
  organisation: ~
  person: James Paledown
 address: '12, place des zics'
  zipcode: 75014
  city: Paris
  country: France
  phone: +33.145147005
  fax: ''
  email: admin@opidah.com
tech-c:
….
bill-c:
….
```

Ces informations permettent à l'attaquant de connaître la victime et ainsi utiliser ces informations comme une piste pour affiner et pousser encore plus loin les recherches. L'attaquant pourra donc avoir quelques détails utiles pour entreprendre une attaque physique ou, dans notre cas, pour étendre le domaine de recherche des failles grâce aux principaux DNS. Le Listing 1. montre un résultat du WHOIS pour *opidah.com*

Il existe des services de consultation des données WHOIS comme ce site par exemple : *http://whois.domaintools.com/*. En utilisant n'importe quel moteur de recherche, vous devriez retrouver des services équivalents, vous évitant les lignes de commandes.

### Protection des informations

Pour protéger vos informations, il y a deux solutions : soit entrer de fausses informations, les rendre privées, soit entrer de fausses informations et rendre un domaine privé.

La première solution est bonne mais vous décrédibilise auprès des clients et il est difficile d'avoir confiance en une entreprise envoyant de fausses informations. La seconde se fait, entre autres, en achetant un nom de domaine qui comprend l'option de type WhoisGuard (*http://www.whoisguard.com/*), rendant vos informations privées comme indiqué au Listing 2.

## Les services de noms

Chaque adresse IP est associée à un serveur DNS qui se charge de renvoyer l'adresse IP. Grâce au transfert de zone, l'attaquant liste les entrées DNS d'un domaine pour s'en servir comme vecteur d'attaque potentiel. Il faut noter que, très souvent, d'anciens serveurs physiques ou virtuels abandonnés sont listés et aident ainsi l'attaquant selon les failles présentes sur ces serveurs.

Sous Unix et Windows (pas sur Linux), un transfert de zone peut se réaliser grâce à la commande `nslookup` comme représenté au Listing 3.

En revanche, sous Linux, il faudra se contenter d'une autre commande puisque les options *nslookup* ne sont pas interfacées comme pour Windows et Unix. La commande à utiliser pour lister tous les *records* permis, sera *dig* comme ceci :

```
wwitb~# dig @<Serveur DNS> <nom_de_
        domaine> axfr
```

## Restreindre le transfert de zone

Après avoir obtenu la liste des entrées, l'attaquant n'aura qu'à tester les différentes adresses et énumérer les services. Comme indiqué précédemment, si un ancien serveur ou un serveur virtuel incorrectement maintenu s'y trouve, l'attaquant l'utilisera contre la victime.

Toutefois, limiter le Transfert de Zone sous Windows s'effectue par l'utilitaire `MMC`, puis sous `Service et Application → DNS → <SERVER> → Forward Lookup Zone → <Nom de Zone>`, puis Propriété et en cochant l'option « `Only to the following servers` » en précisant la liste des serveurs de sauvegarde, ou tout simplement, décochez l'option « `Allow Zone transfert` » si vous jugez ne pas en avoir besoin.

Sous Unix et Linux avec Bind, nous pouvons définir des listes d'accès et restreindre les transferts de zones grâce à la directive allow-transfer comme sur le Listing 4 avec la liste d'accès *tpeux* en modifiant le fichier /etc/bind/named.conf.local.

### Retreindre les requêtes DNS

Un serveur DNS permet à un client de se connecter et effectuer des requêtes concernant les domaines hébergés. Cependant, quelques *open resolvers*, souvent être utilisés pour des attaques DdoS, acceptent les requêtes récursives pour n'importe quel domaine. Les requêtes récursives et non-récursives doivent être interdites pour les clients externes afin d'éviter les *spoofing conditions* ou le *cache*

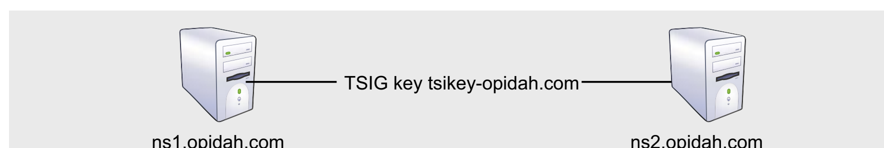



**Figure 1.** *Communication entre deux serveur DNS*

*snooping*. Nous pouvons donc restreindre les requêtes récursives et non-récursives dans une liste d'accès appelée « trusted » comme indiqué dans le Listing 5.

Cette configuration aura pour effet de ne retourner aucun résultat comme souhaité.

## DNS et Encryption : TSIG et DNSSEC

Pour résoudre les problèmes liés *hijacking*, *poisoning* et la sécurité des transferts de zones, nous allons voir comment signer les messages DNS avec une empreinte numérique calculée à l'aide d'une clé secrète partagée entre l'émetteur et de récepteur (voir Figure 1).

Pour le TSIG, nous générerons en exemple une clé tsigkey-opidah.com associée au domaine *opidah.com* :

wwitb˜# dnssec-keygen -a HMAC-MD5 -b 512 -n HOST tsikey-opidah.com.

La génération engendre deux fichiers :

- Ktsikey-opidah.com.+157+43377.key
- Ktsikey-opidah.com.+157+43377.private

Nous pouvons maintenant spécifier la clé générée à notre fichier de configuration `named.conf` :

```
key  tsikey-opidah.com. {
       algorithm hmac-md5;
       secret ''lHctNAZhIlqyekBbIk99
          jghEJg60syXJFAtkBeDOJMNZVP
          eEwM8LtRUr OI6n5Ta2OgmiX1/
          eyWB6EcbglnF5iQ=='';
}
```

Puis nous appliquerons la signature des DNS pour un serveur spécifique :

```
server 192.168.1.7 {
           keys { tsikey-opidah.com.;
          };
}

zone ''opidah.com'' {
        type  master;
        file   ''/var/cache/bind/
          opidah.com.hosts'';
        allow-query { any; };
        allow-transfer { key tsikey-
          opidah.com.,; };
}
```

La communication entre le serveur primaire et secondaire est maintenant vérifiable à travers leurs signatures. Le TSIG est très simple à mettre en place, mais présente quelques désavantages : il faut distribuer les clés parmi les serveurs, il n'y a pas de niveau d'autorité et donc cette technique n'est pas aussi flexible que du cryptage asymétrique.

Pour palier ce problème, le protocole DNSSEC a été proposé. Ce protocole utilise le cryptage en clé publique pour la signature des fichiers de zone.

Pour le DNSSEC, nous pouvons utiliser une commande similaire comme pour le TSIG. Nous allons donc générer une clé DSA de 1024 bits :

```
wwitb~# dnssec-keygen -a DSA -b 1024
          -n ZONE opidah.com.
```

Il faudra par la suite ajouter la ligne précisant le chemin de la clé `$include /Users/fluxius/Kopidah.com.+003+18366.key` à la fin du fichier de configuration de zone. Puis, nous allons générer notre nouveau

**Listing 2.** *WHOIS Protégé avec WhoisGuard*

```
Registrant Contact:
  WhoisGuard
  WhoisGuard  Protected
  45f454e5f888.protected@whoisguard.com
  +1.1256023578
  Fax: +1.1256023578
  11400 W. Olympic Blvd. Suite 200
   Los Angeles, CA 90064
   US
```

**Listing 3.** *Transfert de zone*

```
wwitb~# nslookup
    > ls -d opihad.net
        [ns3.opidah.com]
        $ORIGIN opidah.com.
        @    IN    SOA     ns2.opidah.com. master.opidah.com. (
                       1105498423
                       10800
                       3600
                       604800
                       38400 )

opidah.com.            IN   NS          ns1.opidah.com.
opidah.com.            IN   NS          ns2.opidah.com.
opidah.com.            IN   NS          ns3.opidah.com.
opidah.com.            IN   A            11.22.33.44
www.opidah.com.    IN   CNAME  opidah.com
dev.opidah.com.       IN   CNAME  opidah.com
smtp.opidah.com.     IN   A            22.33.44.55
ftp.opidah.com.        IN   CNAME  opidah.com
mail.opidah.com.      IN   A            22.33.44.55
monsite.com.           IN   MX  10   mail.opidah.com.
ns1.monsite.com.      IN   A            11.22.33.44
```

**Listing 4.** *Restriction des transferts de zone avec Bind*

```
# /etc/bind/named.conf

acl ''tpeux'' {
      192.168.1.0/24;
      172.15.0.0/16;
}

options {
      …
      allow-transfer { tpeux; };
      …
}
```

fichier de zone avec les empreintes (Voir résultat en listing 6.) :

```
wwitb~# dnssec-signzone -o opidah.com
        db.opidah.com
```

## Informations sur les services

### Énumération et fingerprinting

L'énumération permet à l'attaquant de découvrir les services qui tournent sur un serveur et qui sont accessibles en public. Ce genre d'informations sont principalement collectées par un *scan* de ports, permettant à l'attaquant en question de voir quelles attaques peuvent être réalisées contre une organisation par exemple.

Le *fingerprinting* est une étape importante car l'attaquant a besoin d'informations sur les services et modules utilisés sur le serveur cible. Cette étape est suivie, en général, d'une recherche de failles potentielles en utilisant des exploits référencés ou en cherchant soit-même les failles.

L'outil Nmap, comme nous le voyons sur le Listing 7, nous permet d'énumérer tout en capturant les bannières.

Après ce *scan*, nous pouvons inventorier ces informations en dressant un tableau de services (Tableau 3) :

Il ne faut pas oublier aussi de noter le système utilisé qui n'est autre que `Debian` avec une version de noyau comprise entre `Linux 2.6.13 et 2.6.27`. Nous observons, en revanche, qu'il manque des informations concernant l'accès FTP. Donc, nous allons remplacer ce « ? » en faisant une capture de bannière sous `telnet` en s'identifiant comme utilisateur `Anonymous` :

```
220 ProFTPD 1.3.0 Server (Debian) [::
        ffff:11.22.33.44]
USER Anonymous
500 USER not understood
USER root
331 Password required for root.
root
500 ROOT not understood
…
```

Puisque l'accès est protégé, l'attaquant ne pourra pas compter sur l'utilisateur `Anonymous` pour avoir ne serait-ce qu'un petit accès au serveur avec `ftp`.

Le service présent sur le port 8080 semble très intéressant : c'est un accès protégé au routeur. Cependant, la protection en question est très faible car si nous lancions une attaque par *Bruteforce*, il ne tiendra qu'à la faible robustesse du mot de passe pour que l'attaquant puisse y avoir accès rapidement. A savoir que si l'attaquant en question à un accès au routeur, il pourra injecter son propre *firmware* pour espionner les communications qui transitent à travers le routeur (voir Figure 2).

La phase d'énumération et de *fingerprinting* contribue largement à l'identification des vecteurs d'attaque potentiels. En effet, nous sommes partis d'une simple adresse pour avoir son adresse IP, les serveurs DNS associant chaque entrée à une adresse IP et l'énumération avec *fingerprinting* nous permet de connaître les services utilisés en nous aidant à affiner le domaine de recherche des vulnérabilités.

## Prévenir contre l'énumération et le fingerprinting

L'énumération est une étape assez dangereuse car l'attaquant saura où et comment attaquer grâce aux traces laissées par ces services. Bloquer les ports qui ne nous sont pas utiles en extérieur est la première chose à faire avant de mettre

## Pour plus de sécurité

Il est préférable de poser des conditions supplémentaires concernant l'accès au protocole SSH. En effet, plus nous isolerons les services ne regardant pas les autres usagers, moins nous aurons à faire dans le cadre de la maintenance de notre serveur.

Seules sont acceptées ici les connexions au protocole SSH provenant d'une adresse spécifique :

```
wwitb~# iptables -A INPUT -p tcp -s <IP de Management> --dport 22 -m state --
state ESTABLISHED -j
wwitb~# iptables -t filter -A OUTPUT -p tcp -s <IP de Management> --dport 22 -m
state --state NEW, ESTABLISHED -j ACCEPT ACCEPT
```

Il sera toutefois recommandé d'utiliser le *firewall* côté routeur de base en plus pour assurer un filtrage par défaut.

**Listing 5.** *Restriction des requêtes DNS*

```
# /etc/bind/named.conf.local

...
acl ''trusted'' {
      192.168.1.0/24;
      ''Liste d'IPs'';
}
…

options {
      …
      allow-transfer { tpeux; };
      allow-recursion { trusted; };
      allow-query { trusted; }
      version    ''What What!'';
      …
}

zone ''opidah.com'' {
      type master;
      file "/var/cache/bind/opidah.com.hosts";
      allow-query { any; };
      …
}
```

un serveur en ligne. C'est ce que nous verrons avec `IPTables/NetFilter`.

## IPTables/NetFilter

Nous avons ici un *firewall* intégré en natif à Linux à partir du noyau 2.4 mais encore très peu utilisé. Dans notre cas, nous aurons seulement besoin des services HTTP, FTP, SSH, c'est-à-dire ouvrir les connexions entrantes des ports 80 (ou 8080), 21 et 22.

Tout d'abord, il nous faudra réinitialiser les IPTables actuelles :

```
wwitb~# iptables -t filter -F
        wwitb~# iptables -t filter
        -X
```

Ensuite, nous fermerons toutes les connexions entrantes et sortantes :

**Listing 6.** *Fichier zone signé*

```
opidah.com.    38400   IN SOA  ns1.opidah.com.
                       master.opidah.com. (
                    1105498423 ; serial
                    10800      ; refresh (3 hours)
                    3600       ; retry (1 hour)
                    604800     ; expire (1 week)
                    38400      ; minimum (10 hours 40 minutes)
                    )
            38400   RRSIG   SOA 3 2 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CBCh4i2hJ4MEBZOwnrTbBH8qhfmIL0hgXxcS
                    Npzhe2NWVViP/EnW38Y= )
            38400   NS  ns1.opidah.com.
            38400   NS  ns2.opidah.com.
            38400   NS  ns3.opidah.com.
            38400   RRSIG   NS 3 2 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CILFAkABqfTKR3TfcLlPhm1tojcXXB4JGs4K
                    ZQNlaz6gBW2O108DOac= )
            38400   A   11.22.33.44
            38400   RRSIG   A 3 2 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CfcIN53KXkgxVCXxw+VrmRCMyJxwdX6itb8E
                    ODbUBJ/i0L+CUFHZSRE= )
            38400   MX  10 mail.opidah.com.
            38400   RRSIG   MX 3 2 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CCQaBt4tB4T88p1Sh3VuLWuK/3ZDKdSBlCa/
                    vKhnazXpM4lBzCeqhzY= )
            38400   NSEC    dev.opidah.com. A NS SOA MX RRSIG
                       NSEC DNSKEY
            38400   RRSIG   NSEC 3 2 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CfiwxrDyo07TQ7xLapZFjj7YEY1smU4z1YDY
                    GB052jlEa+0AmSgLvJc= )
            38400   DNSKEY  256 3 3 (
                    Ckvc1reoYB/b0AB5qZIbQOLU1fBJn+gDVEXJ
                    YXHZnsJ0O5Z92IMYFEP83UtpF+IuQAZ8F/jf
                    mh1Px0GFLZ9b0/Ba09FAPvwbRGvZ55Eg5JEG
                    z3E04k+sNPoQb6THSIQB3rx2xKorOi1iLKdY
                    7DlXyRyhnl7EHbwA1Q9z+ZPtzsRLkB5LRSOm
                    A4nTk5Cl+IrdSuovsUciMt+wKerR7fDekONY
                    sIIOqjkfkujJ/kYx7ZA2HFh/YmrM0c1Uul8K
                    +49DY94HbP9ftr+xWvNCXxpxOHIwWNE4toYT
                    pAzRRi+0A8NhDCkulkGQOZR5Tnim+Uwymi+i
                    sIn7IsYfGlfkyiQABQMFDpZZALbpcCrxf8PW
                    Fmb6QB1O6Qqvj8kIR5fXiIBczQJc7BpLkftD
                    VSCqIhFFns8wRPiVywHlaPs4ZkXWiEZk8Yxo
                    UgzZFNZ93EnSAyCmJm0gPoyI5zc6t/8DhCMc
                    mUiYIxMGnLClTP/0Q3VfVmTSMiNMUjcd+R3R
                    8rOYTmNRUOKJPJxvNkvOEZ0Q5goa+YhVFFhV
                    ) ; key id = 18366
            38400   RRSIG   DNSKEY 3 2 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CFRmf5NRxWnmuwMXdFNc0H7M8qCRNW3yWzAZ
                    cOfjWoudoHDcAcKL8G4= )
dev.opidah.com.        38400   IN CNAME opidah.com.opidah.com.
            38400   RRSIG   CNAME 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CJCfh1wsCccZNLnepysi94Ab5nThE3LqFioS
                    MQ1/oerKg/u7UUpJlFg= )
            38400   NSEC    ftp.opidah.com. CNAME RRSIG NSEC
            38400   RRSIG   NSEC 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CHHwZ6uk+s8dCIQgVLfbUelUCFPVSoPWBYFF
                    OuOWQIo4L8lW3LFozVA= )
ftp.opidah.com.        38400   IN CNAME opidah.com.opidah.com.
            38400   RRSIG   CNAME 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CcwyPPjuXz0NUlfQkKfHNgexsMK5F2AeEVLC
                    849Sph291NDw9Ny8Xww= )
            38400   NSEC    mail.opidah.com. CNAME RRSIG NSEC
            38400   RRSIG   NSEC 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CF3+skOwfVhvuj+OTNwXxSlMzDullcLvG/fh
                    pZsjNAbktPp4RKixItQ= )
mail.opidah.com.    38400   IN A    22.33.44.55
            38400   RRSIG   A 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CCIZh1AChDwB2uc+HFU+D2Fnt/ElD02Y/igJ
                    T7NHjfqAtpJVJg4NUc4= )
            38400   NSEC    ns1.opidah.com. A RRSIG NSEC
            38400   RRSIG   NSEC 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    ChnFhbXeK8/5m22yU9nboH3uD5TnnA5zI1Wl
                    KAaqgsalPIn8dMbIFV8= )
ns1.opidah.com.        38400   IN A    11.22.33.44
            38400   RRSIG   A 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    Chw+NpxFjNVM/GuzXMHSJY9yTfEakW6AuauZ
                    /AV3A090nyNAQoU+x1I= )
            38400   NSEC    smtp.opidah.com. A RRSIG NSEC
            38400   RRSIG   NSEC 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    ChiaNeMbcHZcL6llahVvMCrLscP3RcddFaAx
                    FhWmbAguq78w8fvk95o= )
smtp.opidah.com.    38400   IN A    22.33.44.55
            38400   RRSIG   A 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CJXouL8pfBwp8jDXWWDa0SWmPHmDNkeSpXxt
                    Nzovv6uEpzNtkQbE5fc= )
            38400   NSEC    www.opidah.com. A RRSIG NSEC
            38400   RRSIG   NSEC 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CJKCaWC1TQxnQojF7Bf/MhPn6aDBGamrbZ2T
                    kCBFMr5JdjGcIvfI4KI= )
www.opidah.com.        38400   IN CNAME opidah.com.opidah.com.
            38400   RRSIG   CNAME 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CBYB8UZtCN/VHz5IXQRFAHhG/VALpAX/lmjq
                    pHoZ8Rj5ZkbDXuqnh00= )
            38400   NSEC    opidah.com. CNAME RRSIG NSEC
            38400   RRSIG   NSEC 3 3 38400 20100121002739 (
                    20091222002739 18366 opidah.com.
                    CbjbHO20U8z4fSHSCqYjpN2y54ajZcDBqBcP
                    0bXmfXtgWjmSdiHfnMc= )
```

```
wwitb~# iptables -t filter -P INPUT
        DROP
wwitb~# iptables -t filter -P FORWARD
        DROP
wwitb~# iptables -t filter -P OUTPUT
        DROP
```

Maintenant, nous allons ouvrir les connexions entrantes et sortantes des services SSH, HTTP et FTP, avec bien évidement un suivi de connexion pour se protéger des *scan* à l'aide d'un port source précis (option -g nmap) :

```
wwitb~# iptables -t filter -A INPUT -p
        tcp --dport 22 -m state --
        state ESTABLISHED -j ACCEPT
wwitb~# iptables -t filter -A OUTPUT
        -p tcp --dport 22 -m state
        --state NEW, ESTABLISHED -j
        ACCEPT
wwitb~# iptables -t filter -A INPUT -p
        tcp --dport 80 -m state --
        state ESTABLISHED -j ACCEPT
wwitb~# iptables -t filter -A OUTPUT
        -p tcp --dport 80 -m state
        --state NEW, ESTABLISHED -j
        ACCEPT
wwitb~# iptables -t filter -A INPUT -p
        tcp --dport 21 -m state --
        state ESTABLISHED -j ACCEPT
wwitb~# iptables -t filter -A OUTPUT
        -p tcp --dport 21 -m state
        --state NEW, ESTABLISHED -j
        ACCEPT
```

Notre politique de filtrage est très restrictive, le *firewall* bloque le protocole ICMP. Pour l'autoriser, il nous faut ajouter ces deux lignes :

```
wwitb~# iptables -A OUTPUT -p icmp -j
        ACCEPT
wwitb~# iptables-A INPUT -p icmp -j
        ACCEPT
```

Pour s'assurer que tout ce qui n'est pas dans les critères est effectivement bloqué :

```
wwitb~# iptables -A INPUT -j DROP
wwitb~# iptables -A OUTPUT -j DROP
```

## Le Signature Apache

Apache à tendance à afficher beaucoup trop de choses pendant les messages d'erreurs dont le type de système d'exploitation, les modules installés et plein d'autres informations concernant les versions utilisées.

Mais, heureusement, il est possible de les enlever grâce à la directive `ServerSignature` dans le fichier *apache2.conf* :

```
ServerSignature On
```

### Le listing des répertoires et fichiers

Le listage des fichiers et de répertoires est une fonctionnalité que l'attaquant utilise contre nous, en voyageant dans les dossiers que nous aimerions tenir à l'égard des regards indiscrets. Pour le désactiver, il nous faudra configurer la directive `Options` comme suit *apache2.conf*:

```
Options -Indexes
```

### Les liens symboliques

Il est nécessaire aussi que l'attaquant ne puisse pas suivre les liens symboliques de fichiers sensibles comme */etc/shadow* (contenant les mots de passes de la machine) et */etc/passwd* :

```
Options -FollowSymLinks
```

### Le Server-Side includes injection

Afin d'éviter les vulnérabilités de type *Server-Side injection*, comme par exemple `<!--#exec cmd="ls"-->`, nous ajouterons un *tag* supplémentaire à notre directive `Options` :

**Tableau 1.** *Services opidah.com*

| Ports | Services | Versions et modules | Autres informations |
|---|---|---|---|
| 443 | open_ssl + httpd | Linksys WRT54G wireless-G router http config | Service https demandant une authentification ldap/htaccess |
| 111 | rpcbind | | \| 100000 2 111/udp rpcbind<br>\| 100024 1 32769/udp status<br>\| 100000 2 111/tcp rpcbind<br>\|_ 100024 1 36381/tcp status |
| 8080 | open_ssl + httpd | Linksys WRT54G wireless-G router http config | Service https demandant une authentification ldap/htaccess |
| 22 | ssh | OpenSSH 4.3p2 Debian 9etch3 (protocol 2.0) | ssh-hostkey: 1024 4c:e2:5f:86:e4:d0:c9:0d:45:6a:95:01:96:6c:ff:cc (DSA)<br>\|_ 2048 cc:1f:19:f1:2c:741:55:0f:45:7e:0a:02:cc:ed:b6:c1 (RSA) |
| 21 | ftp | ProFTPD 1.3.0 | ? |
| 80 | Apache | Apache httpd 2.2.3 ((Debian) DAV/2 SVN/1.4.2 PHP/5.2.8-0.dotdeb.1 with Suhosin-Patch) | Accès public : `html-title:` Opidah – Super Service |
| 143 | imap | Courier Imapd (released 2005) | imap-capabilities: THREAD=ORDEREDSUBJECT QUOTA THREAD=REFERENCES UIDPLUS ACL2=UNION SORT ACL IMAP4rev1 IDLE NAMESPACE CHILDREN |

**Listing 7.** *Scan et fingerprinting de opidah.com*

```
wwitb~# nmap -vA opidah.com

Starting Nmap 5.00 ( http://nmap.org ) at 2009-12-22 12:34 Paris, Madrid
NSE: Loaded 30 scripts for scanning.
Initiating ARP Ping Scan at 12:34
Scanning opidah.com [1 port]
Completed ARP Ping Scan at 12:34, 0.33s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 12:34
Completed Parallel DNS resolution of 1 host. at 12:34, 0.03s elapsed
Initiating SYN Stealth Scan at 12:34
Scanning opidah.com [1000 ports]
Discovered open port 111/tcp on opidah.com
Discovered open port 113/tcp on opidah.com
Discovered open port 21/tcp on opidah.com
Discovered open port 22/tcp on opidah.com
Discovered open port 143/tcp on opidah.com
Discovered open port 80/tcp on opidah.com
Completed SYN Stealth Scan at 12:34, 0.03s elapsed (1000 total ports)
Initiating Service scan at 12:34
Scanning 7 services on opidah.com
Completed Service scan at 12:34, 6.05s elapsed (7 services on 1 host)
Initiating OS detection (try #1) against opidah.com
NSE: Script scanning opidah.com.
NSE: Starting runlevel 1 scan
Initiating NSE at 12:34
Completed NSE at 12:35, 6.11s elapsed
NSE: Script Scanning completed.
Host opidah.com is up (0.00s latency).
Interesting ports on opidah.com:
Not shown: 993 closed ports
PORT    STATE SERVICE    VERSION
21/tcp  open  ftp        ProFTPD 1.3.0
22/tcp  open  ssh        OpenSSH 4.3p2 Debian 9etch3 (protocol 2.0)
|  ssh-hostkey: 1024 4c:e2:5f:86:e4:d0:c9:0d:45:6a:95:01:96:6c:ff:cc (DSA)
|_ 2048 cc:1f:19:f1:2c:741:55:0f:45:7e:0a:02:cc:ed:b6:c1 (RSA)
80/tcp  open  http       Apache httpd 2.2.3 ((Debian) DAV/2 SVN/1.4.2 PHP/5.2.8-
                       0.dotdeb.1 with Suhosin-Patch)
|_ html-title: Opidah - Super Service
111/tcp open  rpcbind
|  rpcinfo:
|  100000  2    111/udp  rpcbind
|  100024  1  32769/udp  status
|  100000  2    111/tcp  rpcbind
|_ 100024  1  36381/tcp  status
113/tcp open  ident
143/tcp open  imap       Courier Imapd (released 2005)
|_ imap-capabilities: THREAD=ORDEREDSUBJECT QUOTA THREAD=REFERENCES UIDPLUS
                       ACL2=UNION SORT ACL IMAP4rev1 IDLE NAMESPACE CHILDREN
MAC Address: 00:11:D8:22:11:00 (Asustek Computer)
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.13 - 2.6.27
Uptime guess: 82.077 days (since Thu Oct 01 11:44:43 2009)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=205 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OSs: Unix, Linux
```

**Listing 8.** *Affichage des erreurs avec PHP*

```
<?php
                       error_reporting(E_ALL);
                       ini_set("display_errors", 1);

 ...
?>
```

```
Options -Indexes -Includes
```

## Éviter les attaques de type Denial of Services sur Apache

Limiter le temps mort et les requêtes permet d'éviter les attaques de Type DoS (*Denial of Service*). L'idée d'une telle protection est de laisser le moins d'espace aux attaquants comme suit :

```
Timeout 30
LimitRequestBody 524288
LimitRequestFields 20
LimitRequestFieldSize 8190
LimitRequestLine 8190
```

Bien plus que le *SYN Flooding, le Ping of Death, Teardrop, Ping Flooding, Amplification Attack, etc...* il existe des scripts tout comme Slowloris (*http://ha.ckers.org/slowloris/*) mettant en évidence les attaques DoS rendant certains services inaccessibles. Cependant, il existe une solution ç des attaques de ce type comme HTTPReady.

Exécution d'une attaque Slowloris :

```
perl slowloris.pl -dns example.com
```

## PHP : Affichage des erreurs

Les utilisateurs n'ont pas souvent conscience qu'afficher les erreurs est très dangereux, surtout quand elles renseignent parfaitement sur la nature d'une erreur de compilation, du fichier contenant l'erreur et de son chemin.

En effet en identifiant l'erreur, l'attaquant pourra alors savoir quel type d'attaque utiliser : Failles Include ou *SQL Injection*.

Pour ce faire, évitez d'afficher les erreurs en modifiant le fichier *php.ini* comme ceci :

`display _ errors Off`

Vous laisserez donc les développeurs s'occuper d'une partie développement pour chaque script avec l'affichage des erreurs comme dans le Listing 8.

## Éviter les attaques Bruteforce avec Fail2ban

Fail2ban est un script qui surveille les accès réseau à l'aide des *logs* du serveur. Un utilisateur tentant de se connecter, selon un nombre de fois qui peut être limité, est banni durant une certaine période.

http://www.fail2ban.org/wiki/index.php/Main_Page

Nous configurons dans le fichier *jail.conf* les services à monitorer :

## Sur Internet

- *http://www.modsecurity.org/documentation/modsecurity-apache/2.5.11/html-multipage/* – ModSecurity,
- *http://whois.domaintools.com/* – Whois domain Online Tools,
- *http://www.bind9.net/manuals* – Manuel d'utilisation Bin9,
- *http://linux.die.net/man/1/dig* – Manuel DNS lookup Linux,
- *http://www.alsacreations.com/tuto/lire/622-Securite-firewall-iptables.html* – Securiser son serveur avec IPTables/Netfilter, Fail2Ban et Rkhunter,
- *http://www.gelato.unsw.edu.au/lxr/source/net/ipv4/tcp_ipv4.c* – Source du TCP/IPv4 Linux,
- *https://www.defcon.org/images/defcon-16/dc16-presentations/defcon-16-pilosov-kapela.pdf* – Stealing The Internet : Documentation extra sur le MITM.

## Veille technologique sur ses services

La veille technologique est une tâche très importante car elle permet de détecter une vulnérabilité avant quelle puisse avoir une conséquence sur notre système.

Pour cela, consultez toutes les sources concernant les services que vous utilisez (blogs, forums, *newsgroups*, Flux RSS, Sites, ...). N'hésitez pas à faire des audits et à utiliser des programmes comme Nessus à la recherche de vulnérabilités même si ce type de programme ne référence pas plus de 40% des failles connues. De plus, n'oubliez pas la lecture des *logs* de chacun de vos services et à mettre régulièrement à jour vos services.

Voici quelques sites fréquemment actualisés référençant des sources d'exploits :

- *http://www.milw0rm.com*
- http://www.securityfocus.com
- http://www.zerodayinitiative.com
- http://labs.idefense.com/
- http://www.packetstormsecurity.org/

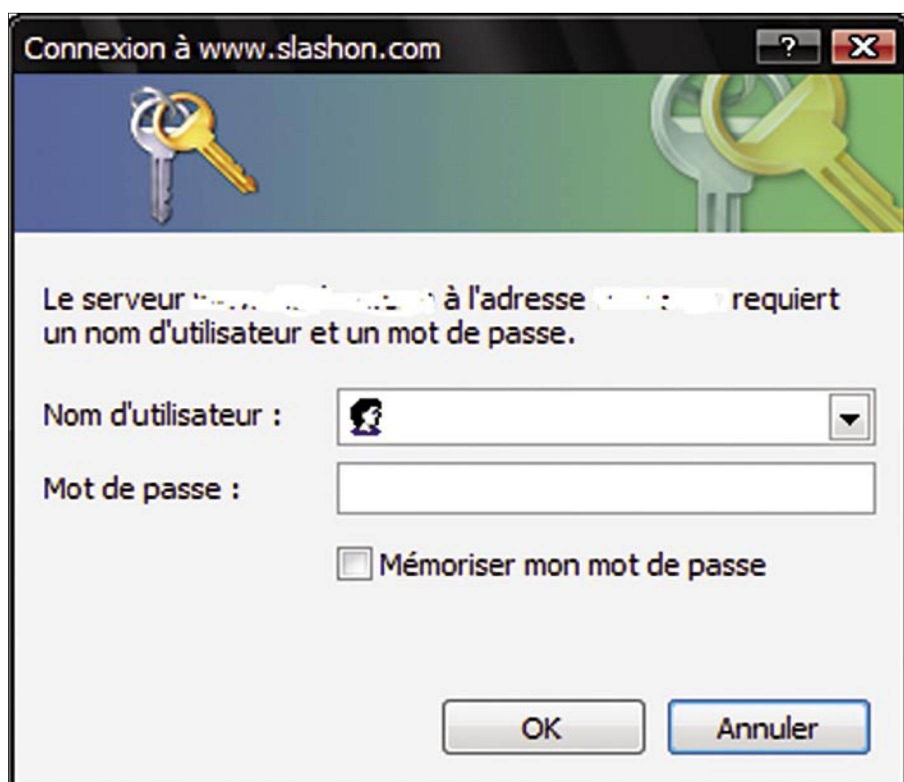



**Figure 2.** *Demande d'identification du routeur*

```
ignoreip = 127.0.0.1  ; liste des
          adresses IP de confiance à
          ignorer par fail2ban
bantime = 600  ; temps de ban en
          secondes
maxretry = 3  ; nombre d'essais
          autorisés pour une
          connexion avant d'être
          banni
destmail admin@opidah.com  ; adresse
          e-mail destinataire des
          notifications
...
```

### Détection des Rootkits

Pour la détection de Rootkits, nous pourrons utiliser `Rkhunter` qui est un anti-Rootkit calculant les empreintes MD5 des programmes afin de détecter les changements chaque jour :

http://rkhunter.sourceforge.net/

## Conclusion

Les serveurs dédiés ne sont pas à l'abri des attaques. Et même si ces règles devraient être appliquées pour chaque poste, les serveurs sont les plus vulnérables car exposés 24h/24, tout peut arriver. En effet, l'attaquant aura le temps de prendre un maximum d'informations pour envisager une attaque ciblée.

Nous n'avons vu qu'une petite partie des attaques variant selon la spécialité de l'utilisateur malveillant et il faut penser à voir le système dans son ensemble. Pensez donc à faire des veilles technologiques, mises à jour des services et avoir d'autres opinions quant à la sécurité de votre serveur.

**À propos de l'auteur**

Étudiant ingénieur en électronique, informatique et management à l'école EFREI, Sébastien Dudek travaille depuis plus de 10 ans dans le domaine de l'informatique dont presque 6 ans dans la sécurité de l'information. Il s'intéresse plus particulièrement aux techniques d'intrusions, failles applicatives, Reversing, Tracking et effectue, ces derniers temps, des recherches dans le domaine Hertzien et la Cryptologie.

Contact : *sdudek.redac@gmail.com*